解　説

論　理　メ　モ　リ*

飯　塚　　肇**

1. はしがき

1.1 論理メモリとは

表題の“論理メモリ”(Logical Memory) という用語は耳なれないことばであり，その定義がなされている文献も筆者は知らない．しかし，ここでは“メモリの各ビット，或はその集合ごとに何等かの同一の論理機能を付加し，単なる情報の記憶のみならず，メモリ上での論理操作を高速に（通常並列に）行なえるようにした記憶装置”の総称として用いることにする．この種のメモリ関連の用語としては，連想記憶(associative memory)，論理付き記憶 (Logic-in-memory)，機能メモリ (functional memory)，分布論理記憶(Distributed logic memory) 等がこれまで用いられているが，この各項については後に触れる．

本解説は，これら論理メモリの歴史をさかのぼり，その背景を調べた後，これまでの代表的アイデアやシステムを概観し，新しい世代の計算機に対するその意義を考察することを目的とする．

1.2 歴　　史

論理メモリの研究の歴史は古く，既に 20 年にも達する．第1のタイプの論理メモリは，連想メモリである．連想メモリは最初に着目され，その金物自身の実現法とシステム構成についての研究が進められた．詳しくは2章に述べるが，その研究は今でも続き，半導体による小規模のメモリが市販されているし，応用システムもアメリカでいくつかの試みがあり，特殊計算機としての価値を見い出しつつある．しかしながら，当初いわれた完全なアソシアティブ計算機の実現は困難な状態にあるといわねばならない．

こうした連想メモリの発展の中で，1962 年に BTL の C. Y. Lee は，若干の記憶とロジックを兼ねそなえたセルを一次元的に結合した新しい型の論理メモリの提案を行なった．これを分布論理記憶 (Distributed Logic Memory；以後 DLM) と呼んでいるが，以後，BTL を中心に改良，拡張研究が進められ，レーダデータ処理用の特殊計算装置，PEPE(Parallel Element Processing Ensemble) に至っている．本解説では論理メモリの第2のタイプとして取り上げる．

第3のタイプの論理メモリは SRI を中心に 1960 年代の後半，盛んに研究された機能メモリ (Functional Memory) である．〔SRI ではロジック付き記憶(Logic-in-Memory；以下 LIM) と呼んでいる．〕この研究はいわゆるセルラロジックから発展したもので，各セルは同一構造で，1ビットの記憶と若干のロジックをもち，全体として，単なる記憶の他，各種の論理操作ができるよう工夫したものである．いろいろの特殊アレイが考え出されたが，実用に供されてはいない．

この他，最近ではこうした研究を背景に，連想処理を中心としたいくつかの実験システムの研究が行なわれ，急速な半導体技術の進歩と共に，論理メモリの実用化のきざしが見え始めている．第5章で代表的な試みについて触れる．

2. 連想メモリ[1),2)]

連想メモリは．この他“内容アドレスメモリ”(Content Addressable Memory；CAM) とか“データアドレスメモリ” (Data Addressed Memory)，また，探索メモリ (Search Memory) 等，多数の別称を持っている．この装置の性質をあらわすためには“連想”等というあいまいな表現よりも，これら別称の方がより的確ではあるが，適当な日本語訳がないので，ここでは慣習にしたがって“連想メモリ”と呼ぶ．

2.1 連想メモリの原理

通常のメモリでは，そこに記憶された情報にアクセスするためにはその情報の存在する位置を指定してやらねばならない図-1(a)(次頁参照)．これに対し，連想メモリでは必要な情報の一部（これを検索タグと呼ぶ）を指定して，残りの部分を探索し，取り出すこと

---

* Logical Memory——A Survey——by Hajime IIZUKA (Computer Division, Electrotechnical Laboratory)

** 電子技術総合研究所　電子計算機部

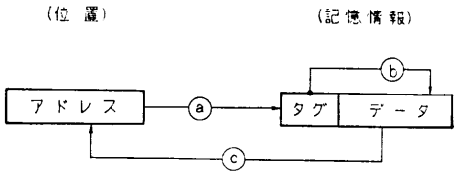

図-1　情報のアクセス関係

がその最大の特徴であって(**図-1(b)**)，それがCAM等の多くの別称のいわれとなっている．

ところで，上記の連想過程はメモリの内容を指定してそのアドレスを取り出し，次に，そのアドレスを用いて，残りの情報を取り出すという2つのプロセスに分解することができる．後者は通常の位置アドレシングに他ならないから，連想メモリの根本は前者の“記憶内容を指定し，そのアドレスを取り出す”**図-1(c)**ことになって，いわば通常のメモリと相対関係にある装置と考えることもできるだろう．

さて，一般的な連想メモリでは記憶語のビットの内，検索タグとして任意ビット位置の組み合わせを指定することができる．この検索位置指定情報は，通常マスクと呼ばれているが，固定されている場合も多い．例をあげよう．**図-2**のように8語×8ビットの連想メモリがあったとする．図において(“00001010”)というマスク #1が与えられたとすると，ビット4，と6がデータと一致する(すなわち00)記憶が検索され，語(1)が読み出されるし，マスク2(“11000001”)なら同じデータでも語(5)が読み出される．

この場合，出力の出し方としては，

① 条件を満足した語の内容全部が読み出される
(マスク1の例では“11100101”)

② 条件を満足した語の位置が与えられる
(例では“001”)

③ すべての語について条件満足かどうかが2値情報として与えられる
(マスク1の例では“01000000”)

④ 一致した語があったかどうかの2値情報(さらにその拡張として，一致なし，一致1個，一致複数の3値情報)

という4つの方法があるが，既に述べたようにその本質は変らない．なお，上記例では検索データとして，記憶語と同じ大きさのものを与えたが，マスクパターンが固定なら検索されないビットを供給する必要は無論ない．

ビット位置 / 語番号	0	1	2	3	4	5	6	7
(0)	1	1	0	1	1	0	1	1
(1)	1	1	1	0	0	1	0	1
(2)	0	0	1	1	1	1	1	0
(3)	0	0	0	0	1	0	0	0
(4)	1	1	1	1	1	1	1	1
(5)	0	1	0	1	1	0	1	0
(6)	1	1	1	1	1	0	0	1
(7)	0	1	0	0	1	0	1	1
検索データ	0	1	1	1	0	0	0	0

(a) 連想メモリの記憶内容とデータ

ビット位置 / マスク番号	0	1	2	3	4	5	6	7
(1)	0	0	0	0	1	0	1	0
(2)	1	1	0	0	0	0	0	1
(3)	1	1	0	0	0	1	0	0

(b) マスクパターン

図-2　連想メモリの動作説明図

次に，連想メモリ構造の一つの問題点を述べる．検索に対し，複数個の語で一致が起こった場合(例えば図-2でマスク 3=“11000100”で検索すれば(5)と(7)が一致する)の処理である．最も単純に適当なアルゴリズムで，その内どれかを優先して取り出してしまうこともできるが，最初のタグで未指定の部分を指定して，くり返し検索し，分離することもでき，いろいろなアルゴリズムが検討されている．特に，出力が原始的な④の形の時は，いずれにしろ内容を知るにはタグの未指定部分を変えてのくり返し検索が必要である．また，①，②の場合は固定的に選択する以外は原則的に多重一致はないものとしなければならない．

2.2 連想メモリのハードウェア

以上述べたように，連想メモリの原理は単純なものであって，我々自身の頭脳での記憶アクセス法を考えてみれば容易に“連想”し得るものである．もし，このようなメモリが，通常のメモリと変わらない速度，コストで得られるならば，計算機の処理能力を飛躍的に高めることができるのは，目に見えているのであるが，現実にそのような計算機が存在しないのは，大容量化した時の検索速度とコストに満足できる連想メモリをハードウェア的に製作することが困難なことによっている．

連想メモリのハードウェアの研究は約 20 年程前からスタートした．1956 年に発表された Slade と Mc Mahonの論文[3]を嚆矢として，クライオトロンを用いたものが数多く研究された．その中には Rosin による計算機システム構成の研究[4]もあり，1965 年には 5 k語×72 ビットのものまで試作された[5]が，結局のところ実用化するには至らなかった．クライオトロンはその小型，低価格（いずれも当時として）論理機能との組み合わせの容易さ等の特性が連想メモリとして適当であると考えられたわけだが，現実には極低温という使いにくさ，高速化が難しいという難点のために消え去ったのである．

クライオトロン以外では磁気コアと半導体が素子として用いられた．前者は 1961 年以来，いくつかの試みがあるが，非破壊読み出しでないと検索を速く行なえないので，読み出しによる内容の破壊の問題をいかに避けるかが大きな問題であり，クライオトロン同様，実用化に至っていない．同じ磁性メモリでも最近研究が進んでいる磁気バブルメモリでは，記憶と論理を一体として組み合せることが容易なので，再び脚光を浴びる日があるかもしれないが，今のところ未知数である．

最後の半導体素子によるものは，原理的には各記憶フリップフロップに一致検出回路を付加すればよいので，論理との組み合わせ，速度の点で最も好都合な方式である．1963 年，E. S. Lee は離散素子によって，9語×8ビットの連想メモリを試作した[6]が，その後 LSI 技術が進歩するにおよんで市販品も発売され，現在実用化されている連想メモリは，殆んど半導体素子によるものである．例えば，Intel 3104 は 4 語×4 ビットのショットキー TTL の高速メモリに連想機能を付加したもので，電源を除いたピン構成は図-3 のようになっており，次の 3 つのモードで動作する．

（1） 読み出し：アドレス線（A_0～A_3）のいずれかを選択すると出力線（O_0～O_3）に対応する番地の 4 ビットが現れる．

（2） 書き込み：アドレス線のいずれかを選択し，WE 信号を入れると入力線（D_0～D_3）に与えられた入力データが対応番地のビットエネーブル線（E_0～E_3）が“1”のビットに対し書きこまれる．

（3） 連想モード：（2）において WE 信号を入れなければ，ビットエネーブル線が“1”のビットに対し，データ線の内容が全記憶語と並列に比較され，一致がとれた語に対し，マッチ信号が M_0～M_3 に 2.1 節に述べたタイプ ③ の出力形式で与えられる．

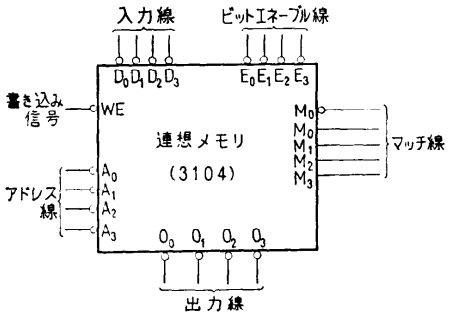

図-3 半導体連想メモリ (Intel 3104) の構成

これらの動作は最大遅延時間 30 ns で行なわれ，出力は開放コレクタ形式なので，複数個組み合せて使用することができる．

この連想メモリは容量が小さいので，大容量システムの実現は困難であるが，後述する仮想記憶のアドレス変換や，キャッシュメモリのアドレス検索等小容量ですむシステムには有効に使われている．最近の半導体技術の進歩は著しいものがあるので，需要に応じさらに大容量のものが近い将来実用化されるであろう．

2.3 連想メモリの応用

連想メモリの本質的利点は，比較操作が全記憶語に対して並列に行なわれ，検索時間が，原理的には記憶語数によらず一定であるところに存在する．したがって，表の検索等の特定条件を満足するデータの検索に多くの応用を見い出すことができ，極めて小規模のものからシステム全体の構成まで各種の提案がある．最近は，比較的大規模な特殊システムの試作例があるが，これについては 4 章で述べることとし，ここでは計算機システム内に実用化されている例を 2，3 あげるにとどめる．

（1） 仮想記憶システムにおけるアドレス変換

仮想記憶方式[7]はユーザの用いる論理アドレス空間とハードウェアの持つ物理アドレス空間を分離し，ユーザに主記憶の物理的制限をあまり意識させないですませる等，利用者側にとって非常に都合のよい特徴を持っている．しかしながら，これを実現するためには各アクセスごとにセグメント表や，ページ表等を参照し，ユーザの論理アドレスを物理アドレスに変換せねばならない．これらの表は通常，主記憶中に格納されているので，毎回それを参照していれば1回のデータアクセスごとに 2〜3 回の主記憶アクセスを行なわねばならない．

ところが，プログラムが一時期にアクセスするアドレスは，普通，集中しているという事実を利用して，極く最近アクセスされた論理アドレスと物理アドレスの対応関係だけを連想メモリに入れておき，各アクセスではまず，この連想メモリを論理アドレスをタグとして検索して物理アドレスを求め，ここにない時だけセグメント表やページ表を参照することにすると，アドレス変換の高速化を期待することができる．シミュレーションや実測の結果，8〜16 語の連想メモリがあれば殆んどのアクセスで主記憶中の表を参照しなくてすむことがわかっており，比較的低コストでパフォーマンス上の大きな利点が得られるので，Multics や IBM 360/67 等で実用に供されている．

（2） キャッシュにおけるマッピング高速化

キャッシュ[8]は高速のメモリを主記憶のバッファとして CPU 中に用意し，実効的に主記憶の速度を向上させる技法であるが，キャッシュの割当てに関し，連想メモリが利用されることがある．すなわち，キャッシュを 1kバイト程度の大きさのセクタに分割し，キャッシュの各セクタごとに1語の連想メモリを用意して，現在そこに割りあてられている主記憶アドレスを入れておく．そして，CPU がアクセスしようとするアドレスで連想サーチを行ない，そのアドレスのデータが記憶されているキャッシュのセクタ番号を高速に求めるわけである．セクタ数は 8，16 程度なので，これと同語数の連想メモリが，IBM 360/85，通産省の超高性能計算機（H-8800）で利用されている．

（3） 記号アドレスの変換[9),10)]

プログラムには実数，整数，アレイ（デスクリプタ）等，いろいろの“名前のついた情報”が用いられる．Manchester 大学の MU-5 では，これらの名前つき情報は1つのセグメントに入っているが，MU-5の処理装置には 64 語のネーム記憶という特別の装置があって，そのアクセスの高速化に利用されている．ネーム記憶の半分は普通のレジスタであるが，残りは連想メモリで，最近アクセスされた名前つき変数はここへ残される．通常の計算機では頻繁に使用される情報はプログラマ自身が，レジスタに割りあてねばならないが，MU-5 では連想メモリによって自動的に行なわれ，実行の高速化，プログラムの容易化に役立っているわけである．

3. 機能メモリ

機能メモリ（Functional memory）という用語は，固有名詞的に用いられることもあるが，ここではメモリの各記憶ビットに適当なロジックを付加して，単なる記憶だけでなく，各種の論理機能を行なえるようにしたものをさすことにする．この意味では連想メモリも含まれるわけであるが，ここでは連想機能のみでなく，もう少し広範な機能を持つものを紹介する．なお，この種の装置を論理的にいろいろ研究した SRI のグループはこれをロジック付き記憶(Logic-In-Memory : LIM) と呼んでいる．

機能メモリの特徴，利点としては次のような項目があげられている．

① 論理的柔軟性：一つのアレイで何種かの操作ができる場合が多い．
② 試験と診断の容易さ：一般ロジックより規則性があるから．
③ 実装密度を高くし，高速動作可能：ファインファンアウトが少なく，並列動作可能．
④ 論理設計容易．
⑤ 回路設計，製造が容易．
⑥ 故障の回避：プログラミングによって，故障セルを避けることが可能な場合もある．
⑦ 素子の技術進歩に容易に適合可能：全システムの改変を要しない．

これらの特徴が生かされれば大変好都合な上，LSI向きであるし，今後が大いに期待されるわけだが，今のところは紙の上の研究が殆んどである．問題点としては，素子の冗長が大き過ぎること，汎用性がどこまで得られるか不明であること等があげられよう．

以下，興味ある2，3の提案を紹介する．

3.1 A C A M[11)]

SRIでは各種の LIM が研究されたが，ここではその中でも比較的汎用性のある機能メモリ ACAM (Augmented Content Addressed Memory) の構成を紹

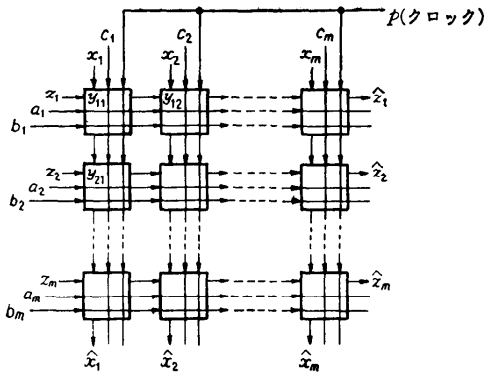

図-4　ACAM の構成（y：フリップフロップの値）

介する．

ACAM の構成は**図-4** に示したようになっていて，各セルは1ビットの記憶（y）とセルに入る信号を操作するロジックから成っている．共通信号として横方向に2本 (a, b)，縦方向に1本 (c) と，全セルに入るクロック信号（p）があるほか，縦方向に伝播しながら各セルで操作を受ける信号 x と横方向に伝わる信号 z がある．各セルが出力する x，z 信号は x，z 入力と共通入力 (a, b, c) と記憶ビットの値 (y) で定まる．**表-1** はその動作表である．y はクロックが入った時のみ変化するから，クロックを初期セット以外に用いなければ“ACAM”はプログラマブルな組み合せ回路で，例えば**図-5** に示したような回路を同じアレイから作ることができる．

表-1　ACAM のセルの動作式

$c\ b\ a$	FF(クロック)y'	水平方向 z	垂直方向 x	動　作
0 0 0	y	$z(x+y)$	x	Product; ⊇Test
0 0 1	y	$z(x\oplus y)$	x	〃 ; =Test
0 1 0	y	$yz+yx$	$yx+yz$	Permutation
0 1 1	y	z	$x+yz$	Nil; Sum (+)
1 0 0	$zy+zx$	z	$x\oplus yz$	Enter; Complement; Sum (⊕) parity check
1 0 1	$zy+zx$	z	y	Read; Shift down
1 1 0	$\bar{z}y+xz$	$\bar{z}z+xy$	x	Bit read; shift left
1 1 1	$y\oplus x\oplus z$	$M(y, x, z)$	x	Add, Sub; unate term; ≧ >test

y を記憶ビットとして用いるメモリ動作では外部から与える信号の組み合わせによって，通常のメモリとしてはもちろんのこと，スタック，キュー，インデクスレジスタ群，スクラッチパドメモリ，ソーティングメモリ（記憶されている語を大きさの順に自動的に並べ変える）等として利用できる．

3.2 “3値”機能メモリ

Mckeever[12]が提案し，イギリス IBM のグループが計算機システムへの応用を研究した[13),14)]機能メモリであって，単に，Functional Memory というとこれをさすこともある．

やはり，連想メモリが基本になっているが，各記憶セルが 0，1，X (don,t care) の3値になっていることが大きな特徴である．ローカル記憶，制御記憶，連想記憶，論理ブロック（組み合せ回路）等として用い

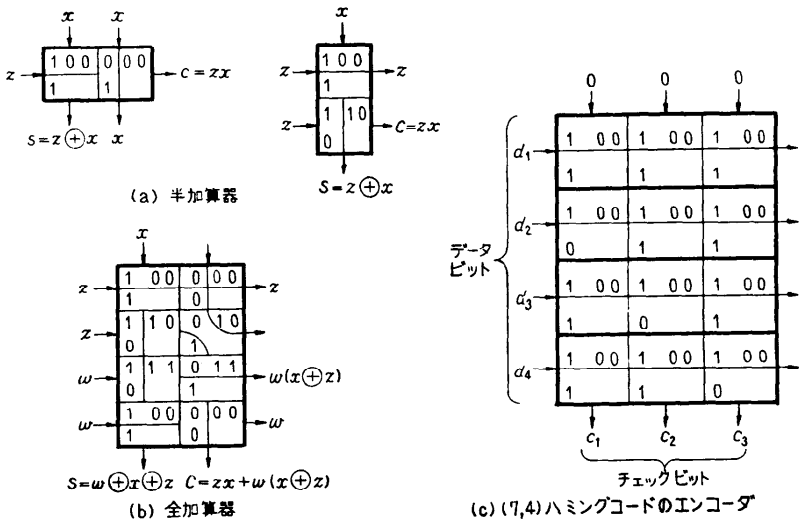

図-5　ACAM の応用例（セル内のビットは $\boxed{\begin{matrix}c\ b\ a\\ y\end{matrix}}$ を表わす）

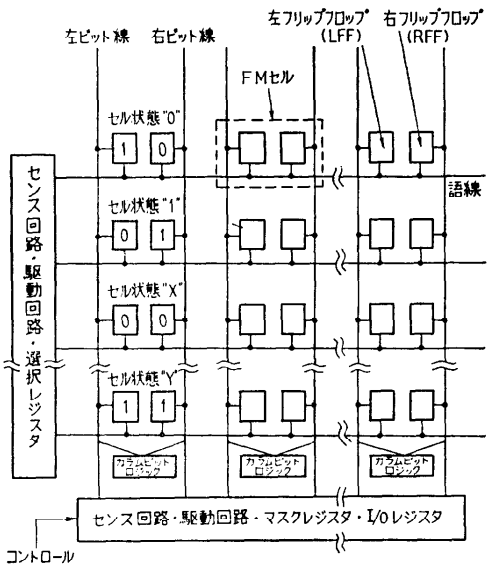

図-6　三値機能メモリの構成

ることができるが，特に，連想機能を用いて，表参照方式で論理関数を実現する時，普通の連想メモリに比べ，語数が著しく少なくてすむ利点がある．

さて，この機能メモリ（以下 FM）は**図-6** に示したように，各記憶ビットが2個のフリップフロップ (LFF, RFF) で構成されたメモリで，状態は図-6 中に示したように割りあてられている．状態 "X"(00) は don't care に用いられ，状態 "Y" (1, 1) はこのメモリの機能に本質的なものではないが，場合により有効に生かせることもある．このアレイの行方向に走る線はその行の全フリップフロップに接続され，語(選択)線を構成する．また列方向のビット（選択）線は各記憶語の同一ビット位置を接続して，各ビット位置ごとのカラムロジックを経て，最終的に I/O レジスタに入る．両者共，センス用，駆動用を兼ねている．

FM の動作は検索フェーズと読み出しフェーズの2つにわかれており，検索フェーズでは**表-2** に従って，各ビット線が駆動され，連想メモリ動作が行なわれる．ここでマスクビットの意味は連想メモリの場合と同じである．各 FF にはそのビット線が駆動された時に，記憶内容が "1" であれば語線に出力 "1" を出す回路が付加されており，同一語線上のその信号は全て OR がとられるので，出口でこれを反転してやると，検索パターンにマッチした語線にのみ出力を得ることができる（但，状態 Y のセルはないものとする）から，これをレジスタ（選択レジスタ）にセットする．

表-2　検索フェーズによる駆動表
(R: 駆動，L: 駆動せず)

I/O レジスタビット	サーチマスクビット	左ビット線	右ビット線
0	1	*L*	*R*
1	1	*R*	*L*
任　意	0	*L*	*L*

また，この選択レジスタはシフトレジスタ構成になっていて，次々と語を読み出す時にシフトして用いられる．

読み出しフェーズは語線が駆動されると，内容が "1" である FF はビット線に出力を出すから，R-ビット線に出力情報が現れる．複数語が駆動された時は語線の場合同様，同一ビット線上の信号は OR がとられる．また，書き込みにはまず，ビット線に書き込みたい値をのせて駆動し，次に選択レジスタのビットが1になっている語線を駆動すればビット線上の値がセルに書き込まれる．

ここでこの機能メモリの特徴を生かした任意関数の生成法を考えてみる．**図-7** は3ビットの情報 A と B のビットごとの任意論理関数を作成する例で，12 語×13 ビットの FM で構成されている．タグフィールドで関数の種類(16 種) を区別しているが，特定の関数に限ればこのフィールドはいらない．図の例は A ="100" と B="010" の排他 OR をとるものであるが，このデータを用いて FM のビット 0～9 につき

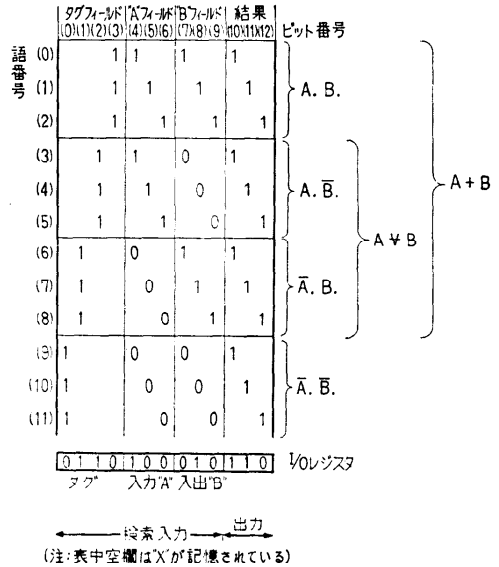

図-7　3ビット語の任意論理関数生成用三値機能メモリ

検索を行なうと，行3と7からマッチ信号が得られ，対応する選択レジスタのビットがセットされる．次いで，これに対応する語選択線を駆動すると結果に対応するビット線（ビット 10～12）に出力 "110" が与えられる．

この FM についてはその後，マイクロプログラム制御の方式[14]や，SNOBOL プロセッサへの適用[15]，独立連想計算機システムとしての研究[16]等，普通の計算機への組み込みの報告が出されているが，ハードウェア的実現が試みられたという報告はない．

3.3 キャッシュの機能メモリ化

キャッシュは最近アクセスされた主記憶中の情報を高速のバッファメモリ（キャッシュ）に入れておき，見かけ上全主記憶が高速化されたように見せる技法であるが，このキャッシュを機能メモリ化すればプログラムには全主記憶が機能メモリ化されたように見せることができるであろう．こうした発想に基づいて，Stone が次のような LIM 計算機の提案[17]を行なっている．

いま，セクタ方式のキャッシュの各セクタを LIM 化し，セクタごとに各種の演算や論理操作がハードウェア的に行なわれるようにすれば，主記憶スペース上での各セクタに対する命令は，実際はこのキャッシュ中でその論理機能を用いて，高速に行なうことができるようになる．命令としては，検索，二つのセクタの対応語の並列加算，一つのセクタ中のデータのスケーリング（すなわち定数を乗じること）各語のビットスライス等の操作が考えられており，うまく実現できれば，アレイデータの処理，記号フィールドの処理等を高速化できる可能性がある．

4. 分布論理記憶

1962 年 C. Y. Lee は新しい形の論理メモリ，分布論理記憶 (Distributed Logic Memory；以下 DLM) を提案[18]した．これは**図-8** のように，若干の記憶とロジックを兼ねそなえたセルを一次元に結合し，各セルに記憶された情報に対し，中央制御装置からの指令に基づいて，主として連想処理を行なうことを目的としたシステムである．Lee の提案以後，いろいろな拡張研究が BTL を中心に行なわれ[19]，セルの機能は次第に複雑になった．近年，BTL を中心に開発されたレーダデータ処理用の連想システム，PEPE (5.3 節) は DLM の研究を基礎としていると考えられる．

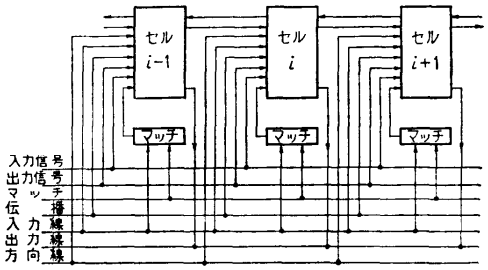

図-8　C. Y. Lee の分布論理記憶

4.1 Lee の DLM[18]

図-8 において，各セルは1字の記憶と若干のロジックで構成され，両隣りとのみ連絡することができる．DLM に記憶される情報は文字列であって，一つの情報は名前とパラメータの対として連続したセルに1字ずつ記憶され，情報（フィールド）相互間や名前とパラメータ間の分離は特殊文字をはさむことにより行なわれる．すべてのセルは中央制御装置の共通制御下にあって，ブロードキャストされる**表-3** に示した4種の基本命令に応答する．この命令は全セルで並列に実行されるが，セルにはアクティブとインアクティブの状態があり，インアクティブ状態のセルでは実行されない．

名前を指定し，パラメータを検索する例をあげよう．

いま，(B, XY)，(AB, XW)，(AC, U) の3個のデータが DLM に記憶されているとすると，α をフィールド分離文字，β を名前とパラメータの分離文字とすれば，各セルの記憶内容は**表-4**（**a**）(次頁参照) のようになるであろう．この時，AB に対するパラメータを検索するには**表-4**（**b**）のようなプログラムで行なうことができる．また，これとは逆に，パラメータから名前を検索することもできるが，この場合は出力過程でアクティビティをその文字例の α まで左へ伝播してもどす必要があるので，ステップ数はやや増加する．

このシステムは並列サーチを行なうので，ステップは多少複雑になるが，物理的条件をぬきにすれば，連想システムの常として検索時間がシステムの大きさによらず高速であること，拡張が容易であるという利点

表-3　Lee の DLM の命令表

入力 (input)	入力線上のデータ（1字）をとりこむ
出力 (output)	出力線へデータを出力する
マッチ (match)	自分のデータを入力線上のデータと比較し，一致すれば隣りのセルをアクティブにする
伝播 (propagate)	アクティブ状態を隣りのセルへ伝播させる

表-4 DLM 中の記憶と検索プログラム例

セル番号	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	⑪	⑫	⑬	⑭	⑮	⑯	⑰
記　号	α	B	β	X	Y	α	A	B	β	X	W	α	A	C	β	U	α

a）DLM 中の記憶状態

① α に対してマッチ（全セル）⟶2, 7, 13 がアクティブになる.
② A に対してマッチ（アクティブのセルのみ）⟶8, 14 がアクティブになる.
③ B に対してマッチ（アクティブのセルのみ）⟶9 がアクティブになる.
④ β に対してマッチ（アクティブのセルのみ）⟶10 がアクティブになる.
⑤ 出力⟶X が出力される.
⑥ 右へ伝播⟶11 がアクティブになる.
⑦ 出力⟶W が出力される.
⑧ 右へ伝播⟶12 がアクティブになる.
⑨ 出力⟶α が出力される. 出力された記号は中央制御装置で監視しているので α が出力されたことで終了がわかる.

b）検索プログラム

がある. しかしながら, 非常に多数のセルを一次元に結合するので, 最も遠いセルでもブロードキャストされた命令の実行が完了することを保証するためには, 1ステップの時間はあまり小さくてできない等実用化にはいろいろ問題があった.

4.2 Crane の DLM[19]

BTL の Crane 等は Lee のアイデアを拡張し, **図-9** のような DLM を提案したが, このシステムでは各セルは全てのデータの同一ビットを記録するように修正されている. 各セルには前と同じく, 一致を記憶するフリップフロップ M と比較, 書き込み, 読み出し等の論理回路があって, 1語の全ビットの並列サーチが可能になっている. 命令の種類も増しているが, 基本的には Lee の場合と同様である.

この DLM は演算速度は向上するが, 入出力は Lee の場合より難かしいので, さらに, それを改良し2つ

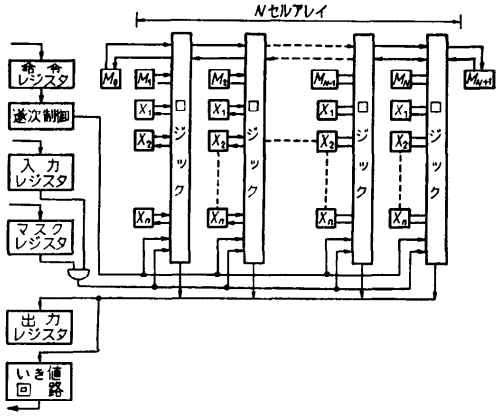

図-9 Crane の DLM

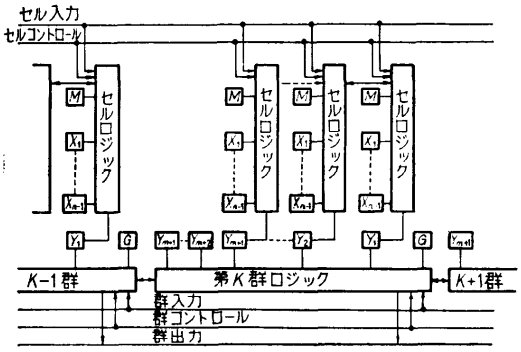

図-10 二次元 DLM

の DLM を組み合わせ, 二次元構成にした DLM (**図-10**)が考え出されている. ただし, この場合は一般性がやや減り, 構造が複雑化してメモリからやや離れ, プロセッサ的色彩が出てくる結果になっている. 当然のことながら, この種の構成はデータがグループ化され, それぞれのデータに多量の類似処理を施す場合に都合がよい.

5. 連想メモリシステム

連想メモリ等の論理メモリを中心に構成したシステムは古くから研究され, 数々の提案がある. その一部は既に各項でふれたが, ここでは実際にハードウェアが製作されたか, 或はそれを前提として考案されたシステムの例を2, 3紹介する.

5.1 STARAN[20),21)]

STARAN は Goodyear aerospace 社で開発された論理メモリ（連想メモリ）をベースとした初の実用計算機システムである. その最初のモデルは 1969 年アメリカ空軍向けに製作されたが, それを改良した STARAN-S が 1972 年以来, 航空管制用等の応用分野で実用に供されている.

システムの構成は**図-11**（次頁参照）に示したブロック図のようになっているが, その中心は 1～32 個の連想メモリアレイで, ここで処理が行なわれる. 各連想アレイは**図-12**（次頁参照）のように 256 語×256 ビットの多次元アクセスメモリアレイと各語に対応する 256 個の処理要素 (PE) で構成されている. このメモリは多次元の言葉が示す通り, 語方向（語スライス）の 256 ビット（一つの語の全ビット）の読み書きの他にビット方向（ビットスライス）の 256 ビット（全部の語の同一ビット）を, 並列に読み出して連想処理のできる論理メモリである. PE の処理はビット直列で

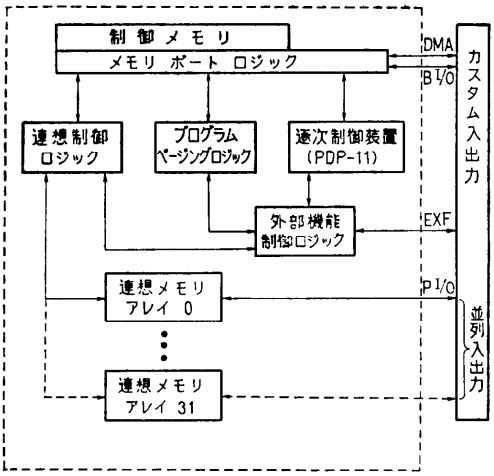

図-11 STRAN のシステム構成

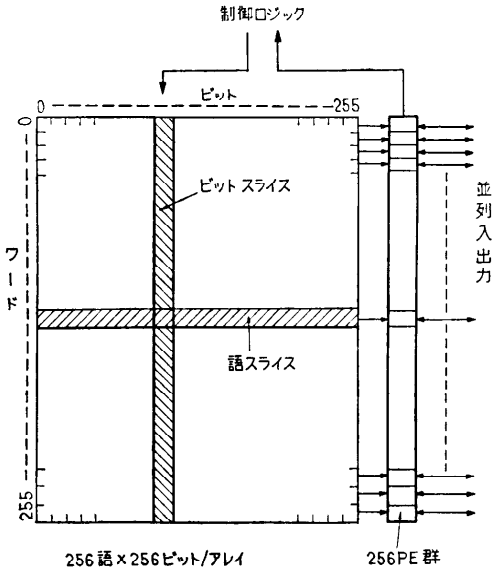

図-12 STARAN の連想メモリアレイ

あるが，選択された語に対してすべて並列に行なわれるから，同時処理すべき語数が多ければスループットは著しく向上する．

したがって，STARAN が有効に働くためにはその応用が次のような特徴を持っていなければならない．

① 非常に高速な（実時間）処理を要求すること．

② 同種の処理を要するデータアイテムの数が多いこと．

③ 高度にアクティブなデータベースの処理と各種の複雑な問合せ（例えば，一致，大小，限界等が複雑に組み合わさった検索と処理）に，短時間で応答する必要がある時，

実際には航空管制(レーダデータの処理)，センサデータの処理，情報検索，テクスト編集，天気予報等の応用が考えられている．

なお，STARAN は図-11 の外部制御論理の代わりに普通の周辺装置をつけて，独立して動作させることもできるが，それぞれのインタフェースを作成し，ホスト計算機と接続して用いることも多い．現在 HIS-645 (Multics) や XDSΣ5 をホストとしたシステムが稼動している．また，STARAN のプログラミングには APPLE (Associative Processor Programming LanguagE) というアセンブラレベルの言語が用意されている[22]．詳細は文献を参照されたい．

5.2 D A P[23]

DAP (Distributed Array Processor) は，イギリスの ICL 社で開発中のアレイプロセッサである．多数の PE をアレイ状に配列し，主制御ユニット (MCU) からブロードキャストされる命令を並列的に実行する点は Illiac IV に類似しているが，各 PE が Illiac IV の場合のような複雑な処理装置でなく 4096×1 ビットのメモリに1ビット幅の簡単な処理ユニットを付加したメモリ中心の構成になっていることが大きな違いである．PE へのデータの記憶法は

a) 主記憶モード（記憶語の各ビットを PE のカラムに割付ける）．

b) アレイモード（一語の全ビットを一つの PE に格納する）．

の2種がある．また，DAP は純然たる主記憶として動作させることもできる．

したがって，この DAP はこれまで述べてきたビットごとにロジックを付加した論理メモリとはやや趣きを異にするものの，4096 ビットごとにロジック（PE 処理ユニット）を付加した一種の論理メモリと解釈することができる．

DAP は現在 LSI 化の試作，シミュレーションによる性能研究が行なわれており，近い将来には 500～30,000 PE のシステムを用いて天気予報等の高度並列計算を行なうことが期待されている．

5.3 その他の論理メモリシステム

既に述べたように論理メモリ（連想メモリ）を用いたシステム構成については，古くから多数の提案があり，単なる提案に終ったものを含めれば数えきれない数にのぼる．しかし，実際にハードウェアの製作が試みられたものは上記2例の他は，DLM から発展した

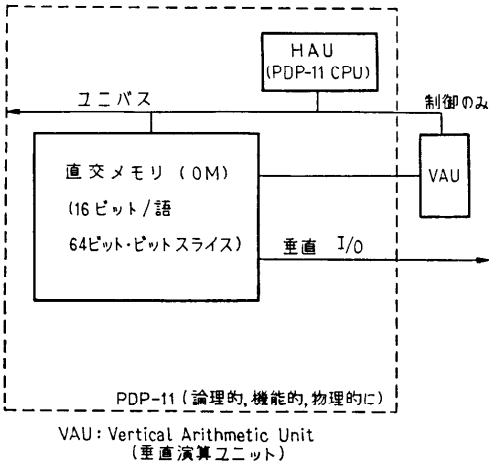

図-13 OMEN-60 連想計算機の構成

特殊連想システム PEPE[24] (Parallel Element Processing Ensemble) が有名である．これは BTL の基本設計をもとに SDC, Honeyuell 等で実用化が進められ，STARAN の場合と同じくミサイル防衛のためのレーダデータ処理等に用いられているが，PE は複雑化し，もはやメモリとはいいがたくなっている．

また，Sanders associates 社が開発した OMEN-60[25] シリーズは，STARAN と同じく，両方向アクセスのメモリ（直交メモリ (orthogonal memory; OM) と呼ぶ）を用いて，連想処理，アレイ処理等（特に FFT 等の信号処理を目的としている）を高速に行なえるよう工夫したものであるが，**図-13** のように PDP-11 のユニバスに接続できるようになっている．図で OM は 8 k～128 k (16 ビット/語) の MOS による論理メモリで，水平演算ユニット (HAU) の PDP-11 からは，ごく普通に語単位で読み書きされるが，垂直演算ユニット (VAU) からのアクセスに対しては同一ビット位置を 64 ビットずつ取り出すことができる．VAU は STARAN 同様，各ビットごとに対応した 64 個の PE を中心に構成され，並列処理が行なわれる．

6. むすび

以上，論理メモリとその応用システムについて概観した．何度もくり返すが，この種の研究は紙の上ではかなり古くから行なわれてきたものの，本格的にハードウェアの製作が試みられるようになったのはここ数年のことである．最近では小規模の連想メモリの応用はほぼ実用化し，STARAN に代表される連想システムの商品化も近づいてきた．半導体技術の進歩と計算機応用の多様化，高性能化の要求を考える時，少くも特殊計算機としてのこうした応用は今後次第に増してくるであろう．

一方，3 章に述べた機能メモリについては，今のところ実用化はあまり進んでいない．これらのものは，① 高密度，② 回路間結合の規則性，③ 内部回路量に比しインタフェース端子が少い，④ 試験が容易等というメモリ固有の規則構造性のために，LSI 向きということがしばしば主張されている．にもかかわらず，未だ，提案のみに終っているのは，構造上いろいろな制限条件が加わることや，汎用性を与えた時，素子の冗長度が著しく大きくなること等にもよるが，半導体技術の著しい進歩によってランダムロジックがメモリの規則構造性に対し，それほど不利にならないことにもよるであろう．このことはランダム論理による LSI ワンチップマイクロプロセッサの最近のすさまじいまでの進歩がよく示している．したがって，こうした機能メモリの将来性は必ずしも楽観を許されないが，連想メモリや直交メモリのように構造の確定したものは次第に大規模なものが LSI 化されていくと考えられる．

参考文献

1) A. G. Hanlon: "Content-addressable and associative memory systems, A survey", IEEE Trans. EC-15, No. 4, pp. 509～521, (Aug., 1966.)
2) 山口: "最近の連想記憶装置について", 電気試験所彙報, Vol. 30, No. 12, pp. 969～991, (Dec., 1966.)
3) A. E. Slade and H. O. McMahon: "A cryotron catalog memory system", Proc. EJCC, pp. 115～120, (Dec., 1956.)
4) R. F. Rosin: "An organization of an associative cryogenic computer", Proc. SJCC, pp. 203～212, (May, 1962.)
5) J. D. Barnard et al: "Structure of a cryogenic associative processor", Proc. IEEE Vol. 52, pp. 1182～1190 (Oct., 1964.)
6) E. S. Lee: "Associative techniques with complementing flip-flops", Proc, SJCC, pp. 381～394 (May., 1963.)
7) 例えば，吉広，黒沢: "バーチャルメモリについて (1, 2)", 情報処理, Vol. 14, No. 9 & 10, pp. 701～706 & pp. 778～785 (1973).
8) 例えば，飯塚: "キャッシュメモリシステム (1, 2)", 情報処理, Vol. 13, No. 7 & 8, pp. 467～473 & pp. 540～547 (1972).
9) D. Aspinall et al: "Associative memories in large computer systems", Proc. IFIP,

Vol. 2, pp. 796～800 (1968).
10) R. N. Ibhett: "The MU 5 instruction pipeline", computer J., Vol. 15, No. 1, pp. 42～50 (Feb., 1972). Proc. IFIP 68, Vol. 2, pp. 796～800.
11) W. H. Kautz: "An augmented content-addressed memory array for implementation with large-scale integration", JACM, Vol. 18, No. 1, pp. 19～33 (Jan., 1971).
12) B. T. McKeever: "The associative memory structure", Proc. FJCC, pp. 371～388 (1965).
13) M. Flinders, et al: "Functional menory as a general purpose systems technology", IEEE Computer group conf., pp. 314～324, (1970).
14) P. L. Gardner: "Functional memory and its microprogramming implications", IEEE Trans. C-20, No. 7, pp. 764～775 (July., 1971).
15) G. E. Rossmann and L. H. Jones: "Functional Memory-based dynamic microprocessor", ACM Proc. SIGPLAN-SIGMICRO interface meeting, pp. 37～65. (May, 1973).
16) M. A. Wesley et al: "A design for an auxilary associative parallel processor", Proc. FJCC, pp. 461～472, (1972).
17) H. S. Stone: "A logic-in-memory computer", IEEE Trans. C-19, No. 1, pp. 73～78, (Jan., 1970).
18) C.Y. Lee: "Intercommunicating cells, basis for a distributed logic computer", Proc. FJCC, pp. 130～136, (1962).
19) B. A. Crane and J. A. Githens: "Bulk processing in distributed logic memory", IEEE Trans. EC-14, No. 2, pp. 186～196, (Apr., 1965).
20) J. A. Rudolph: "A production implementation of an associative array processor-STARAN". Proc. FJCC, pp. 229～241. (1972).
21) K.E.Batcher: "STARAN parallel processor system hardware", Proc. NCC, pp. 405～410 (1974).
22) E.W. Davis: "STARAN parallel processor system software", Proc. NCC, pp. 17～22 (1974).
23) S. F. Reddaway: "DAP-A distributed array processor", Proc. first annual symp. on computer architecture, pp. 61～65 (Dec., 1973).
24) 例えば, B. A. Crane et al: "PEPE computer architecture", COMPCON, pp. 57～60 (1972).
25) L. C. Higbie: "The OMEN computers: associative array processors", COMPCON, pp. 287～290. (1972).
26) 飯塚: "セルラシステム" 計算機システム技術, 第8章. オーム社 (1973).

(昭和50年1月6日受付)